

Formulas for Counting the Sizes of Markov Equivalence Classes of Directed Acyclic Graphs

Yangbo He

HEYB@PKU.EDU.CN

*LMAM, School of Mathematical Sciences, LMEQF, and Center for Statistical Science,
Peking University*

Bin Yu

BINYU@STAT.BERKELEY.EDU

Departments of Statistics and EECS, UC Berkeley

Editor:

Abstract

The sizes of Markov equivalence classes of directed acyclic graphs play important roles in measuring the uncertainty and complexity in causal learning. A Markov equivalence class can be represented by an essential graph and its undirected subgraphs determine the size of the class. In this paper, we develop a method to derive the formulas for counting the sizes of Markov equivalence classes. We first introduce a new concept of core graph. The size of a Markov equivalence class of interest is a polynomial of the number of vertices given its core graph. Then, we discuss the recursive and explicit formula of the polynomial, and provide an algorithm to derive the size formula via symbolic computation for any given core graph. The proposed size formula derivation sheds light on the relationships between the size of a Markov equivalence class and its representation graph, and makes size counting efficient, even when the essential graphs contain non-sparse undirected subgraphs.

Keywords: Directed acyclic graph; Markov equivalence class; Size formula; Causality

1. Introduction

A Markov Equivalence class contains all statistically equivalent models of directed acyclic graphs (DAG) (Pearl, 2000; Spirtes et al., 2001). In general, observational data is not sufficient to distinguish an underlying DAG from the others in the same Markov equivalence class. The size of a Markov equivalence class is the number of DAGs in the class. It plays an important part in papers to measure the “uncertainty” of causal graphs or to evaluate the “complexity” of a Markov equivalence class in causal learning (Chickering, 2002; He and Geng, 2008). For example, He and Geng (2008) propose several criterions, all of which are defined on the sizes of Markov equivalence classes, to measure the uncertainty of causal graphs for a candidate intervention; choosing interventions by minimizing these criterions makes helpful but expensive interventions more efficient. Maathuis et al. (2009) introduce a method to estimate the average causal effects of the covariates on the response by considering the DAGs in the equivalence class; the size of the class determines the complexity of the estimation.

An essential graph represents a Markov equivalence class and its undirected subgraphs determine the size of the class (Andersson et al., 1997). The size of a small Markov equivalence class can be counted via traversal methods that list all DAGs in the Markov equivalence class (Gillispie and Perlman, 2002). Recently, He et al. (2015) propose a size counting algorithm that calculates the size of a Markov equivalence class via partitioning the class recursively. In general, this method is efficient for Markov equivalence classes represented by sparse essential graphs, but becomes much time-consuming when the essential graphs contain non-sparse undirected subgraphs.

Counting graphs based on formulas is usually elegant and efficient. Robinson (1973, 1977) provide recursive formulas to count DAGs with a given number of vertices. Steinsky (2003) develops recursive formulas to count Markov equivalence classes of size 1. Later, Gillispie (2006) introduces recursive formulas for arbitrary size, based on all configurations of the undirected essential graphs that produce this size. However, there are few formulas available for counting the size of a given Markov equivalence class, except five formulas introduced in He et al. (2015) for Markov equivalence classes represented by five specific types of undirected essential graphs (trees, graphs with up to two missing edges, etc.).

In this paper, we focus on the formulas for counting the size of a Markov equivalence class. We first introduce a new concept of “core graph”, which is an undirected chordal graph without dominating vertices. An undirected essential graph can be represented by its core graph and the number of dominating vertices. The size of the corresponding Markov equivalence class is a polynomial of the number of dominating vertices given its core graph. Then we develop an iterative method to derive the polynomial, and give the explicit polynomials for both several specific types of core graphs and all core graphs with up to five missing edges. Based on symbolic computation, we introduce a size formula derivation algorithm and a formula-based size counting algorithm for general core graphs and Markov equivalence classes, respectively. Our experiments show that the proposed size formula derivation is efficient in general and formula-based algorithm can speedup size counting dramatically for the Markov equivalence classes represented by essential graphs with non-sparse undirected subgraphs.

The rest of the paper is arranged as follows. In Section 2, we give a brief introduction about Markov equivalence class and size counting of Markov equivalence classes. In Section 3, we propose a method to derive the size formulas and to count the sizes of Markov equivalence classes based on these formulas. In Section 4, we study the size formulas and formula-based size counting of Markov equivalence classes experimentally. We conclude in Section 5 and finally present all proofs in the Appendix.

2. Markov Equivalence Class and Size Counting

A graph \mathcal{G} consists of a vertex set V and an edge set E . A graph is directed (undirected) if all of its edges are directed (undirected). A sequence of edges that connect distinct vertices in V , say $\{v_1, \dots, v_k\}$, is called a path from v_1 to v_k if either $v_i \rightarrow v_{i+1}$ or $v_i - v_{i+1}$ is in E for $i = 1, \dots, k - 1$. A path is *partially directed* if at least one edge in the path is directed. A path is directed (undirected) if all edges are directed (undirected). A *cycle* is a path from a vertex to itself.

A *directed acyclic graph* (DAG) \mathcal{D} is a directed graph without any directed cycle. Let V be the vertex set of \mathcal{D} and τ be a subset of V . The *induced subgraph* \mathcal{D}_τ of \mathcal{D} over τ , is defined to be the graph whose vertex set is τ and whose edge set contains all of those edges of \mathcal{D} with two end points in τ . A *v-structure* is a three-vertex induced subgraph of \mathcal{D} like $v_1 \rightarrow v_2 \leftarrow v_3$. A graph is called a *chain graph* if it contains no partially directed cycles. The isolated undirected subgraphs of the chain graph after removing all directed edges are the chain components of the chain graph. A *chord* of a cycle is an edge that joins two nonadjacent vertices in the cycle. An undirected graph is *chordal* if every cycle with four or more vertices has a chord.

A graphical model is a probabilistic model for which a DAG denotes the conditional independencies between random variables. A *Markov equivalence class* is a set of DAGs that encode the same set of conditional independencies. Let the *skeleton* of an arbitrary graph \mathcal{G} be the undirected graph with the same vertices and edges as \mathcal{G} , regardless of their directions. Verma and Pearl (1990) prove that two DAGs are *Markov equivalent* if and only if they have the same skeleton and the same v-structures. Moreover, Andersson et al. (1997) show that a Markov equivalence class can be represented uniquely by an *essential graph*, denoted by \mathcal{C} , which has the same skeleton as \mathcal{D} , and an edge is directed in \mathcal{C} if and only if it has the same orientation in every equivalent DAG of \mathcal{D} . An essential graph is a chain graph and each of its chain components is an undirected and connected chordal graph (UCCG for short).

Let $\text{Size}(\mathcal{C})$ denote the size of the Markov equivalence class represented by \mathcal{C} (size of \mathcal{C} for short). Clearly, $\text{Size}(\mathcal{C}) = 1$ if \mathcal{C} is a DAG; otherwise \mathcal{C} may contain at least one chain component, denoted by $\mathcal{C}_{\tau_1}, \dots, \mathcal{C}_{\tau_k}$. We can calculate the size of \mathcal{C} by counting the DAGs in Markov equivalence classes represented by its chain components using the following equation (Gillispie and Perlman, 2002; He and Geng, 2008):

$$\text{Size}(\mathcal{C}) = \prod_{i=1}^k \text{Size}(\mathcal{C}_{\tau_i}). \quad (1)$$

Since each chain component is an undirected and connected chordal graph, to obtain the size of a Markov equivalence class, it is sufficient to compute the size of Markov equivalence classes represented by these UCCGs according to Equation (1).

Let \mathcal{U} be a UCCG, τ be the vertex set of \mathcal{U} and \mathcal{D} be a DAG in the equivalence class represented by \mathcal{U} . A vertex v is a *root* of \mathcal{D} if all directed edges adjacent to v are out of v , and \mathcal{D} is *v-rooted* if v is a root of \mathcal{D} . A *v-rooted sub-class* of \mathcal{U} is the set of all *v-rooted* DAGs in the Markov equivalence class represented by \mathcal{U} . A *v-rooted essential graph* of \mathcal{U} , denoted by $\mathcal{U}^{(v)}$, is a graph that has the same skeleton as \mathcal{U} , and an edge is directed in $\mathcal{U}^{(v)}$ if and only if it has the same orientation in every *v-rooted* DAG of \mathcal{U} . He et al. (2015) show that a *v-rooted sub-class* of \mathcal{U} can be represented uniquely by a *v-rooted essential graph* and a Markov equivalence class can be partitioned into sub-classes represented by its rooted essential graphs.

Lemma 1 *Let \mathcal{U} be a UCCG over $\tau = \{v_i\}_{i=1, \dots, p}$, $\mathcal{U}^{(v_i)}$ be v_i -rooted essential graph, and $f(\mathcal{U}^{(v_i)})$ be the size of v_i -rooted sub-class represented by $\mathcal{U}^{(v_i)}$. We have $\text{Size}(\mathcal{U}^{(v_i)}) \geq 1$ for*

any $i = 1, \dots, p$, and

$$\text{Size}(\mathcal{U}) = \sum_{i=1}^p \text{Size}(\mathcal{U}^{(v_i)}). \quad (2)$$

For any $i \in \{1, \dots, p\}$, the undirected subgraphs of $\mathcal{U}^{(v_i)}$ in Lemma 1 are UCCGs, so we can calculate $\text{Size}(\mathcal{U}^{(v_i)})$ in Equation (2) using Equation (1). As a result, using Equation (1) and Equation (2), He et al. (2015) propose to calculate the size of a Markov equivalence class by partitioning it recursively into rooted sub-classes until the sizes of all these sub-classes can be completely determined by the numbers of vertices and edges. However, when the UCCGs contain non-sparse subgraphs, this method might be much time-consuming.

In the next section, we will show that the size of the Markov equivalence class represented by a UCCG depends on a subgraph of the UCCG, and introduce a size formula derivation algorithm and a formula-based counting algorithm, which can greatly accelerate size counting of Markov equivalence classes with non-sparse undirected subgraphs.

3. Formulas for sizes of Markov equivalence classes

In this section, we introduce the concept of core graph that determines the size formula of a Markov equivalence class in Section 3.1. Then, we discuss the recursive and explicit formulas for the size of a Markov equivalence class given its core graph in Section 3.2. Finally, in Section 3.3, we provide algorithms to derive size formulas and to count the sizes of Markov equivalence classes based on these formulas.

3.1 Core graph

A vertex is *dominating* in a UCCG \mathcal{U} if it is adjacent to all other vertices in \mathcal{U} . A *dominating vertex pruned* subgraph of \mathcal{U} is obtained by removing some dominating vertices from \mathcal{U} . We denote a dominating vertex pruned subgraph of \mathcal{U} as \mathcal{U}^{m-} if it is obtained by removing m dominating vertices from \mathcal{U} . An *extended* graph of H , denoted by H^{m+} , is a graph obtained by adding m dominating vertices to H .

Definition 2 (Core graph of a UCCG) *The core graph of \mathcal{U} is the minimal dominating vertex pruned subgraph of \mathcal{U} .*

Let m be the number of dominating vertices in \mathcal{U} , \mathcal{K} be the core graph of \mathcal{U} . Clearly, \mathcal{K} is the same as \mathcal{U}^{m-} . If \mathcal{U} is a completed graph, all vertices in \mathcal{U} are dominating, so the core graph of \mathcal{U} is a *null graph*. Let \mathcal{K} be an undirected graph over V . Clearly, according to Definition 2, the undirected graph \mathcal{K} is a core graph of some UCCG if and only if \mathcal{K} is an undirected chordal graph without dominating vertices. The *complement* of \mathcal{K} , denoted by \mathcal{K}^c , is a graph on the same vertices and an edge appears in \mathcal{K}^c if and only if it does not occur in \mathcal{K} . Proposition 3 presents a property of the complement of a core graph.

Proposition 3 (Complement of core graph) *Let \mathcal{U} be a UCCG, m be the number of dominating vertices in \mathcal{U} , \mathcal{K} be the core graph of \mathcal{U} , \mathcal{K}^c be the complement of \mathcal{K} . We have that \mathcal{K}^c be a connected graph, and for any two edges in \mathcal{K}^c , either they share a common vertex, or they are connected by an edge.*

This property helps us to construct a core graph. In Table 1, we list all core graphs and the corresponding complement graphs of the UCCGs with up to three missing edges.




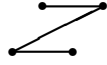






| Number (missing edges) | 0 | 1 | 2 | 3 | | |
|----------------------------|-------------------------|---|---|--|---|---|
| \mathcal{K} | \mathcal{K}_\emptyset |  |  |  |  |  |
| \mathcal{K}^c | \mathcal{K}_\emptyset |  |  |  |  |  |

Table 1: Core graphs and their complements when at most three edges are missing, \mathcal{K} , \mathcal{K}^c , and \mathcal{K}_\emptyset denote a core graph, the complement of \mathcal{K} , and a null graph, respectively.

Let \mathcal{U} be a UCCG with m dominating vertices, \mathcal{K} be the core graph of \mathcal{U} . As an extended graph of \mathcal{K} , \mathcal{U} is the same as \mathcal{K}^{m+} regardless the labels of vertices, so we have $\text{Size}(\mathcal{U}) = \text{Size}(\mathcal{K}^{m+})$. Clearly, the size of the Markov equivalence class represented by a UCCG \mathcal{U} is determined by its core graph \mathcal{K} and the number of dominating vertices m . For an undirected chordal graph \mathcal{K} and a nonnegative integer m , we define a function $f(\mathcal{K}, m)$ as following,

$$f(\mathcal{K}, m) := \text{Size}(\mathcal{K}^{m+}). \quad (3)$$

From the definition of the formula $f(\mathcal{K}, m)$, we have the following lemma directly.

Lemma 4 *Let \mathcal{K} be an undirected chordal graph, and \mathcal{K}^{k+} be an extended graph of \mathcal{K} , we have $f(\mathcal{K}^{k+}, m) = f(\mathcal{K}, m + k)$.*

Consider the UCCGs with at most two missing edges, as shown in Table 1, there is only one core graph exists, so the sizes of the corresponding Markov equivalence classes are determined given the number of vertices in the UCCGs. When three edges are missing in the UCCGs, there are three core graphs exists, so three sizes are possible given the number of vertices. This explains the results introduced in He et al. (2015) that the size of a Markov equivalence class is determined given the number of vertices (p) only when no more than two edges are missing in UCCGs.

The size of \mathcal{U} might be very huge; for a UCCG \mathcal{U} with p vertices, $\text{Size}(\mathcal{U})$ reaches the maximum $p!$ when \mathcal{U} is a completed graph. In general, more edges in the UCCG (more denser), more larger the corresponding class and more time-consuming of size counting. Fortunately, a dense UCCG \mathcal{U} might has sparse core graph \mathcal{K} when many dominating vertices exist. In the next section, given the core graph \mathcal{K} , we will discuss the formula of $f(\mathcal{K}, m)$ that can be used to speedup the enumeration of $\text{Size}(\mathcal{U})$.

3.2 Size formulas based on core graphs

In this section, we propose a method to derive the size formula $f(\mathcal{K}, m)$ defined in Equation (3). We first introduce a recursive formula of $f(\mathcal{K}, m)$ given \mathcal{K} , then propose a method

to derive the explicit size formulas, and finally give the explicit formulas for both several specific types of core graphs and all core graphs with up to five missing edges.

Theorem 5 introduces the main recursive formula for the size of a Markov equivalence class whose representation graph is extended from an undirected chordal graph \mathcal{K} as follows.

Theorem 5 *Let \mathcal{K} be an undirected chordal graph over V . For any integer $m \geq 0$, \mathcal{K}^{m+} is an extended graph of \mathcal{K} , and $f(\mathcal{K}, m)$ is the size of \mathcal{K}^{m+} defined in Equation (3). We have $f(\mathcal{K}, 0) = \text{Size}(\mathcal{K})$, and for any integer $m > 0$,*

$$f(\mathcal{K}, m) = m \cdot f(\mathcal{K}, m-1) + \sum_{v \in V} f(\mathcal{K}_{N_v}, m) \frac{\text{Size}(\mathcal{K}^{(v)})}{\text{Size}(\mathcal{K}_{N_v})}, \quad (4)$$

where $\mathcal{K}^{(v)}$ is a v -rooted graph of \mathcal{K} and \mathcal{K}_{N_v} is an induced subgraph on the neighbors of v .

Theorem 5 shows that the size function $f(\mathcal{K}, m)$ can be calculated through the term $f(\mathcal{K}, m-1)$ and the terms related to some subgraphs of \mathcal{K} . Below, we discuss the explicit formula of $f(\mathcal{K}, m)$. First, we have the following corollary.

Corollary 6 *Let \mathcal{K} be an undirected chordal graph. The formula $f(\mathcal{K}, m)$ defined in Equation (3) is a polynomial divisible by $m!$.*

Consider the recursive formula in Equation (4), the second term in the right side is crucial to derive the explicit formula of $f(\mathcal{K}, m)$. Define

$$g(\mathcal{K}, m) := \frac{1}{m!} \sum_{v \in V} f(\mathcal{K}_{N_v}, m) \frac{\text{Size}(\mathcal{K}^{(v)})}{\text{Size}(\mathcal{K}_{N_v})}. \quad (5)$$

If \mathcal{K} is an undirected chordal graph, its induced subgraph \mathcal{K}_{N_v} is also an undirected chordal graph. According to Corollary 6, the formula $f(\mathcal{K}_{N_v}, m)$ is a polynomial divisible by $m!$, it follows that the formula $g(\mathcal{K}, m)$ defined in Equation (5) is a polynomial of m . Let d be the degree of polynomial $g(\mathcal{K}, m)$, according to Corollary 6, $g(\mathcal{K}, m)$ can be represented by

$$g(\mathcal{K}, m) = \sum_{i=1}^{d+1} \gamma_i m^{i-1}. \quad (6)$$

Given the polynomial $g(\mathcal{K}, m)$, the following theorem shows the explicit formula of $f(\mathcal{K}, m)$.

Theorem 7 *Let \mathcal{K} be an undirected chordal graph, $\{\gamma_i, i = 1, 2, \dots, d+1\}$ be the coefficients of the polynomial $g(\mathcal{K}, m)$ defined in Equation (6), and let $a_{ij} = (-1)^{j-i} \binom{j}{i-1}$ for any $i \leq j$. We have, for any $m \geq 0$,*

$$f(\mathcal{K}, m) = \left(\beta_0 + \sum_{i=1}^{d+1} \beta_i m^i \right) m!, \quad (7)$$

where $\beta_0 = \text{Size}(\mathcal{K})$, $\beta_{d+1} = \gamma_{d+1}/a_{d+1,d+1}$, and $\beta_i = (\gamma_i - \sum_{j=i+1}^{d+1} a_{i,j} \beta_j)/a_{i,i}$, for any integer $i \in [1, d]$.

According to Theorem 7, to obtain the explicit formula of $f(\mathcal{K}, m)$ for an undirected chordal graph \mathcal{K} , we just need to calculate the size $\text{Size}(\mathcal{K})$, and the polynomial $g(\mathcal{K}, m)$ defined in Equation (6). The algorithms for general core graphs \mathcal{K} will be introduced in Section 3.3. Below, we discuss the formulas for some specific types of undirected chordal graphs.

When an undirected chordal graph contains some isolated vertices, these vertices can be removed and the corresponding size formula can be obtained as follows.

Corollary 8 (Isolated vertices) *The graph \mathcal{K} is composed of an undirected chordal graph \mathcal{K}_1 and j isolated vertices. We have*

$$f(\mathcal{K}, m) = f(\mathcal{K}_1, m) + j \cdot \text{Size}(\mathcal{K}_1) \cdot mm!. \quad (8)$$

Epecially, when \mathcal{K}_1 is a null graph, we have $f(\mathcal{K}, m) = (jm + 1)m!$.

A tree is a connected graph without cycle, and a *tree plus* graph is generated by adding one more edge to a tree. We give four explicit size formulas for four specific types of undirected chordal graphs in Corollary 9.

Corollary 9 *Let \mathcal{K} be an undirected chordal with p vertices.*

1. *If \mathcal{K} is a null graph, we have $f(\mathcal{K}, m) = m!$.*
2. *If \mathcal{K} is a tree, we have $f(\mathcal{K}, m) = [(p - 1)m^2 + (2p - 1)m + p]m!$.*
3. *If \mathcal{K} is a tree plus, we have $f(\mathcal{K}, m) = [m^3 + 2pm^2 + (4p - 1)m + 2p]m!$.*
4. *If \mathcal{K} is composed of isolated edges, we have $f(\mathcal{K}, m) = 2^{p/2-1}(pm^2/2 + 3pm/2 + 2)m!$.*

By Corollary 8, corollary 9 and Theorem 7, we can obtain the size formula $f(\mathcal{K}, m)$ given an undirected chordal graph \mathcal{K} . He et al. (2015) give two explicit size formulas for essential graphs with one or two missing edges; here we do the same for core graphs with at most five missing edges. In Table 2, we list all core graphs with up to five missing edges, together with their corresponding size formulas. We give an example to demonstrate the derivation of these formulas. Consider the last (with id 16) core graph in Table 2, \mathcal{K} is composed of a completed graph with five vertices (\mathcal{K}_1) and one isolated vertex. We have $\text{Size}(\mathcal{K}_1) = 120$ and $f(\mathcal{K}_1, m) = (m + 5)!$ from Lemma 4, it follows $f(\mathcal{K}, m)/m! = [(m + 5)! + 120mm!]/m! = 120m + (m + 5) \cdots (m + 1)$ by Corollary 8.

Given an undirected connected chordal graph \mathcal{U} , when its core graph \mathcal{K} is small, we can calculate $g(\mathcal{K}, m)$ directly following its definition in Equation (5), and then obtain the explicit formula of $f(\mathcal{K}, m)$ according to Theorem 7. However, when the core graph is large, the derivation of $g(\mathcal{K}, m)$ becomes more complicated. In the next section, we will provide an algorithm to derive the explicit formulas of $f(\mathcal{K}, m)$ for a general core graph \mathcal{K} .

















| id | (n', p) | \mathcal{K} | $f(\mathcal{K}, m)/m!$ | id | (n', p) | \mathcal{K} | $f(\mathcal{K}, m)/m!$ |
|----|-----------|---|--------------------------|----|-----------|--|----------------------------------|
| 1 | (1, 2) |  | $2m + 1$ | 9 | (4, 5) |  | $24m + (m + 4) \cdots (m + 1)$ |
| 2 | (2, 3) |  | $m^2 + 5m + 2$ | 10 | (5, 4) |  | $m^2 + 7m + 2$ |
| 3 | (3, 3) |  | $3m + 1$ | 11 | (5, 5) |  | $2m^3 + 11m^2 + 29m + 10$ |
| 4 | (3, 4) |  | $3m^2 + 7m + 4$ | 12 | (5, 5) |  | $m^3 + 10m^2 + 19m + 10$ |
| 5 | (3, 4) |  | $m^3 + 6m^2 + 17m + 6$ | 13 | (5, 5) |  | $m^3 + 10m^2 + 19m + 10$ |
| 6 | (4, 4) |  | $4m^2 + 12m + 4$ | 14 | (5, 6) |  | $m^4 + 14m^3 + 55m^2 + 82m + 40$ |
| 7 | (4, 4) |  | $2m^2 + 8m + 3$ | 15 | (5, 6) |  | $(m + 1)(2m + 3)(m^2 + 7m + 16)$ |
| 8 | (4, 5) |  | $(m + 1)(m + 4)(2m + 3)$ | 16 | (5, 6) |  | $120m + (m + 5) \cdots (m + 1)$ |

Table 2: The explicit formulas for all core graphs with up to five missing edges, n', p are the number of missing edges and the number of vertices in the core graph \mathcal{K} , respectively.

Algorithm 1: SizeF(\mathcal{K})

Input: \mathcal{K} , an undirected chordal graph;
Output: $f(\mathcal{K}, m)$, a polynomial of m .

- 1 Let $type$ be the type of \mathcal{K} and p be the number of vertices in \mathcal{K} ;
- 2 **switch** $type$ **do**
- 3 **case** *null graph* **return** $m!$;
- 4 **case** *tree* **return** $[(p - 1)m^2 + (2n - 1)m + p]m!$;
- 5 **case** *tree-plus* **return** $(m^3 + 2pm^2 + (4p - 1)m + 2p)m!$;
- 6 **case** *isolated-edge graph* **return** $2^{p/2-1}(pm^2/2 + 3pm/2 + 2)m!$
- 7 Let w be the number of dominating vertices in \mathcal{K} ; remove these vertices from \mathcal{K} ;
- 8 **if** $w > 0$ **then**
- 9 $h(m) \leftarrow \text{SizeF}(\mathcal{K})$;
- 10 **return** $h(m+w)$
- 11 Let k be the number of isolated vertices in \mathcal{K} ; remove these vertices from \mathcal{K} ;
- 12 **if** $k > 0$ **then**
- 13 **return** $\text{SizeF}(\mathcal{K}) + \text{Size}(\mathcal{K})kmm!$, (see $\text{Size}(\mathcal{K})$ in Algorithm 2);
- 14 **return** $\text{SizeGF}(\mathcal{K})$, (see $\text{SizeGF}(\mathcal{K})$ in Algorithm 1.1);

3.3 Algorithms

In this section, we introduce two main algorithms. The algorithm $\text{SizeF}(\mathcal{K})$ in Algorithm 1 gives the explicit formula of $f(\mathcal{K}, m)$ for an undirected chordal graph \mathcal{K} . The algorithm

Algorithm 2: Size(\mathcal{C})

Input: \mathcal{C} , an essential graph;

Output: the size of Markov equivalence classes represented by \mathcal{C} .

- 1 Let $\mathcal{C}_1, \dots, \mathcal{C}_J$ be all of chain components of \mathcal{U} ; for any integer $0 \leq J \leq J$, m_j is the number of dominating vertices in \mathcal{C}_j and \mathcal{K}_j the core graph of \mathcal{C}_j ;
 - 2 **for** $j \leftarrow 1$ **to** J **do**
 - 3 $f_j(m) \leftarrow \text{SizeF}(\mathcal{K}_j)$;
 - 4 **return** $\prod_{j=1}^J f_j(m_j)$.
-

Size(\mathcal{C}) in Algorithm 2 counts the size of the Markov equivalence class represented by an essential graph \mathcal{C} . Both Algorithm 1 and Algorithm 2 call each other recursively.

In Algorithm 1, we first give the explicit formula of $f(\mathcal{K}, m)$ when \mathcal{K} is null, tree, tree-plus or isolated-edge graph according to Proposition 9. Otherwise, when the undirected chordal graph \mathcal{K} contains dominating vertices or isolated vertices, we simplify the formula derivation according to Lemma 4 or Corollary 8, respectively. Finally, for a general undirected chordal graph \mathcal{K} , we derive the explicit formula of $f(\mathcal{K}, m)$ by the algorithm called SizeGF(\mathcal{K}) in Algorithm 1.1.

The algorithm SizeGF(\mathcal{K}) in Algorithm 1.1 first calculates the polynomial $g(\mathcal{K}, m)$ defined in Equation (6) and then derives the explicit polynomial $f(\mathcal{K}, m)$ according to Theorem 7. Suppose that the undirected chordal graph \mathcal{K} contains J isolated connected subgraphs, we calculate the polynomial $g(\mathcal{K}, m)$ in the first part of Algorithm 1.1 (line 1 to 4) according to Corollary 10 as follows.

Corollary 10 *Let \mathcal{K} be an undirected chordal graph with J isolated connected subgraphs, denoted by $\mathcal{K}_1, \dots, \mathcal{K}_J$ respectively, $V(\mathcal{K}_j)$ be the set of vertices in \mathcal{K}_j , and $g(\mathcal{K}, m)$ is the polynomial defined in Equation (6). We have*

$$g(\mathcal{K}, m) = \sum_{j=1}^J \frac{\text{Size}(\mathcal{K})}{\text{Size}(\mathcal{K}_j)} \sum_{v \in V(\mathcal{K}_j)} \frac{f(\mathcal{K}_{j, N_v}, m)}{m!} \frac{\text{Size}(\mathcal{K}_j^{(v)})}{\text{Size}(\mathcal{K}_{j, N_v})}, \quad (9)$$

where $\mathcal{K}_j^{(v)}$ is the v -rooted essential graph of \mathcal{K}_j , and \mathcal{K}_{j, N_v} is the induced subgraph of \mathcal{K}_j on the neighbours of v .

In Algorithm 1.1, we need to calculate $\text{Size}(\mathcal{K}_j^{(v)})$ for some j and v , which are the sizes of Markov equivalence classes represented by rooted essential graphs. He et al. (2015) propose an algorithm called ChainCom to construct the rooted essential graph and all of its chain components for a UCCG and a root vertex. We give ChainCom in Algorithm 3 in Appendix for the completion of the paper.

In Algorithm 2, we first find the core graphs of the chain components of the essential graph \mathcal{C} , then calculate the size of the corresponding Markov equivalence class by using the formulas obtained from Algorithm 1. When some subgraphs of these chain components contain dominating vertices, formula-based size counting will display its advantages; this will be studied experimentally in the next section.

Algorithm 1.1: SizeGF(\mathcal{K})

Input: \mathcal{K} , an undirected chordal graph;
Output: $f(\mathcal{K}, m)$, a polynomial of m .

- 1 Let $\mathcal{K}_1, \dots, \mathcal{K}_J$ be J UCCGs in \mathcal{K} , $V(\mathcal{K}_j)$ be the vertex set of \mathcal{K}_j ;
- 2 Set $S_{\mathcal{K}_j}^{(v)} \leftarrow \text{Size}(\mathcal{K}_j^{(v)})$ for any integer $j \in [1, J]$ and any $v \in V(\mathcal{K}_j)$;
- 3 $S_{\mathcal{K}_j} \leftarrow \sum_{v \in V(\mathcal{K}_j)} \text{Size}(\mathcal{K}_j^{(v)})$, $S_{\mathcal{K}} \leftarrow \prod_{j=1}^J S_{\mathcal{K}_j}$;
- 4 $g(m) \leftarrow \sum_{j=1}^J \frac{S_{\mathcal{K}_j}}{S_{\mathcal{K}}} \sum_{v \in V(\mathcal{K}_j)} \frac{\text{SizeF}(\mathcal{K}_{j,N_v})}{m!} \frac{S_{\mathcal{K}_j}^{(v)}}{\text{Size}(\mathcal{K}_{j,N_v})}$ and denote it as $\sum_{i=1}^{d+1} \gamma_i m^{i-1}$;
- 5 Set $\beta_0 \leftarrow S_{\mathcal{K}}$; $\beta_{d+1} \leftarrow \gamma_{d+1}/a_{d+1,d+1}$ and $a_{ij} \leftarrow (-1)^{j-i} \binom{j}{i-1}$ for $i \leq j \leq d+1$;
- 6 **for** $i \leftarrow d$ **to** 1 **do**
- 7 $\beta_i \leftarrow \frac{\gamma_i - \sum_{j=i+1}^{d+1} a_{i,j} \beta_j}{a_{i,i}}$;
- 8 **return** $\sum_{i=0}^{d+1} \beta_i m^i m!$.

4. Experimental Results

In this section, we introduce the implementation of the formula derivation and formula-based counting algorithms, and conduct experiments to evaluate the formula-based size counting algorithm proposed in Section 3. All experiments are run on a linux server at Intel 2.0GHz. These experiments display that the proposed algorithms greatly speed up the size counting, especially when the corresponding UCCGs contain dense subgraphs.

4.1 A Python package for size formula derivation

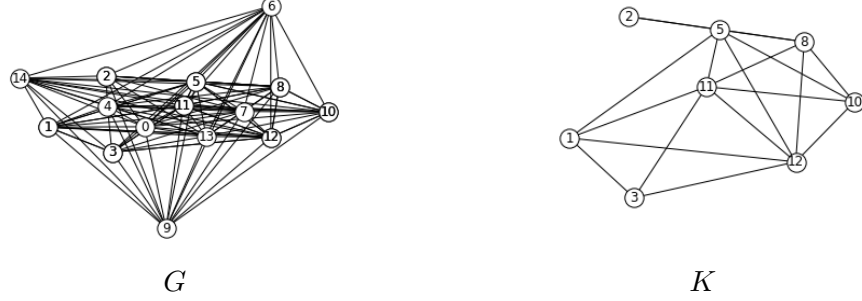
We developed a Python package named countMEC to derive the size formulas and to count the sizes of Markov equivalence classes based on these formulas. The symbolic computation in countMEC depends on the python package *sympy*. The following example demonstrates the usage of the package countMEC.

```

1. from countMEC import *
2. G=ran_conn_chordal_graph(15,95)
3. K=core_graph(G)
4. F=SizeF([K])
5. S=Size(G)

```

In this example, we first import the package countMEC, and randomly generate a UCCG G with 15 vertices and 95 edges. The graph G is shown in the left of Figure 1. Then, we get the core graph of G , denoted by K , which is shown in the right of Figure 1. The graph G contains 7 dominating vertices and the core graph K just contains 8 vertices and 17 edges. In the fourth line, we call $\text{SizeF}(\cdot)$ (Algorithm 1); it outputs the following size formula: $F(m) = (m^3 + 16m^2 + 77m + 108)(m+2)!$. In the last line, we call $\text{Size}(\cdot)$ (Algorithm 2) and get $S = 643749120$, which is the size of G . It's easy to check that $S = F(7)$. In this example, it takes 0.5 second to count size using the proposed formula-based algorithm, while 440 seconds are taken with the method introduced in He et al. (2015); we will compare the time complexities of two methods thoroughly in the next section.


 Figure 1: A UCCG G with 15 vertices and 95 edges and its core graph K .

4.2 Formula-based size counting

In this section, we experimentally compare the time complexity of our proposed counting algorithms to the benchmark algorithm introduced in He et al. (2015). Let \mathbb{U}_p^n be the set of Markov equivalence classes with p vertices and n edges. We obtain random choral graphs from \mathbb{U}_p^n following He et al. (2015). First, we construct a tree by connecting two vertices (one is sampled from the connected vertices and the other from the isolated vertices) sequentially until all p vertices are connected. Then, we randomly insert an edge such that the resulting graph is chordal, repeatedly until the number of edges reaches n . Repeating this procedure N times, we obtain N samples from \mathbb{U}_p^n for each integer $j(\leq n)$.

We first consider the UCCGs in \mathbb{U}_p^n with $p \leq 12$ for each integer $n \in [p+2, p(p-1)/2-3]$. Because the results have the similar patterns for different p , we just report the experiments for $p = 12$ in this paper. Based on the 10^5 samples from \mathbb{U}_{12}^n for each integer $n \in [14, 63]$, we plot the mean, the minimum, the median, and the maximum of the counting time used by the benchmark algorithm (blue dashed lines) and by the proposed Algorithm 2 (red solid lines) in four panels of Figure 2, respectively. In each panel of Figure 2, the main window displays all results ($n \in [14, 63]$) of both algorithms, the two upper sub-windows display the results of both algorithms for $n \in [14, 39]$ and $n \in [40, 50]$, respectively, and the lower sub-window displays the results of Algorithm 2 again with a proper size-coordinate.

We see that the counting time (mean, minimum, median, and maximum) of the benchmark algorithm is increasing in the number of edges (n); size counting based on benchmark algorithm becomes much time-consuming when the graphs are dense. Meanwhile, the time used by Algorithm 2, increases first, and then decreases with the number of edges. Figure 2 shows that size counting based on Algorithm 2 keeps efficient for both sparse and dense graphs.

We also study the sets \mathbb{U}_p^n that contain UCCGs with tens of vertices under sparsity constraints. The number of vertices p is set to 20, 50, and 100, and the number of edges n is set to rp where r is the ratio of n to p . For each p , we consider three ratios: 3, 4 and 5. The graphs in \mathbb{U}_p^{rp} are sparse since $r \leq 5$. For each pair of (p, r) , 10^5 UCCGs are generated randomly and then sorted in ascending order according to the counting time used by benchmark algorithm. The ordered 10^5 UCCGs are divided into four subsets. The subset S_1 contains the first 500 UCCGs, S_2 contains the next 49500 UCCGs, S_3 contains

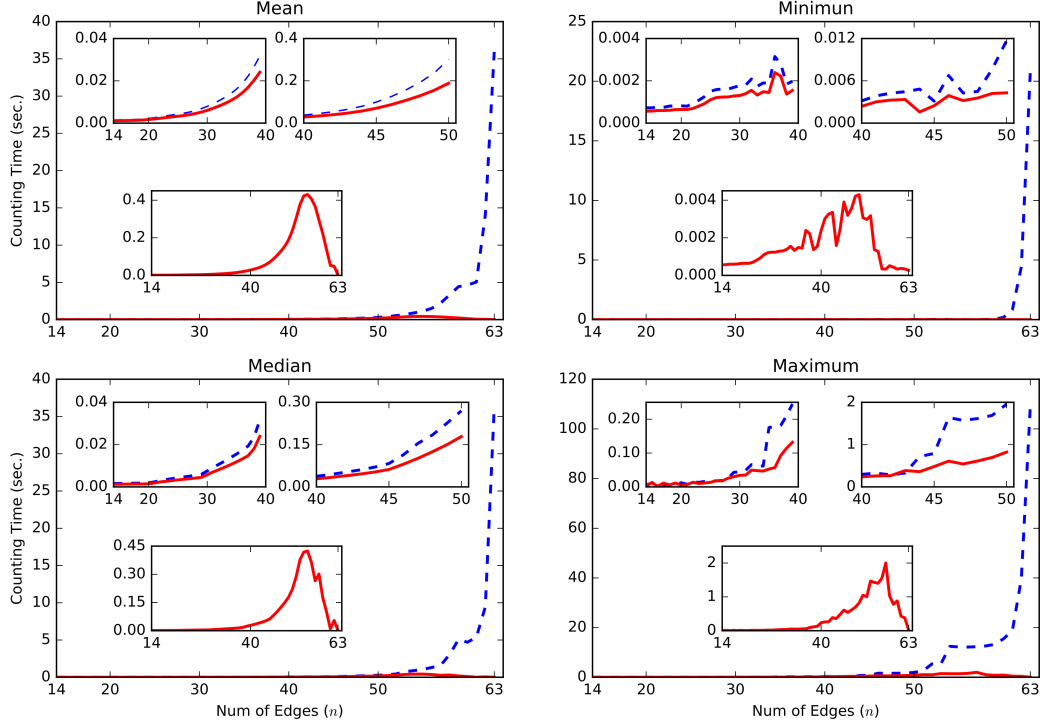


Figure 2: The mean, the minimum, the median and the maximum of counting time of Markov equivalence classes with 12 vertices and n edges.

the next 49500 UCCGs after S_2 , and S_4 contains the last 500 UCCGs. For each subset, we report the average of counting time and the average of their ratios in Table 3 for the benchmark algorithm (T_1) and the proposed algorithm 2 (T_2). We see that on average, (1) the proposed Algorithm 2 is faster than the benchmark algorithm in all cases, (2) the more edges the UCCGs have (r from 3 to 5), or the more time benchmark algorithm used (subset from S_1 to S_4), the smaller T_2/T_1 , that is, the higher speedup Algorithm 2 achieved. For example, consider the subsets S_4 and $r = 5$, the average counting time is shorten rapidly for all $p \in \{20, 50, 100\}$, the average of ratios T_2/T_1 are also reduced to nearly 0.02.

We have to point out that the choral graphs generated in our experiments might not be uniformly distributed in the space of chordal graphs and that the results in Figure 2 and Table 3 are not accurate estimations of expectations of the corresponding statistics.

5. Conclusion and discussion

In this paper, we propose a method to derive the size formulas of Markov equivalence classes and to count the sizes based on these formulas. A core graph of an undirected connected

| p | Subset | r | | | | | | | | |
|-----|--------|-------|-------|-----------|--------|-------|-----------|----------|--------|-----------|
| | | 3 | | | 4 | | | 5 | | |
| | | T_1 | T_2 | T_2/T_1 | T_1 | T_2 | T_2/T_1 | T_1 | T_2 | T_2/T_1 |
| 20 | S_1 | 0.01 | 0.01 | 0.76 | 0.02 | 0.02 | 0.76 | 0.02 | 0.02 | 0.75 |
| | S_2 | 0.03 | 0.03 | 0.77 | 0.17 | 0.13 | 0.74 | 1.47 | 0.96 | 0.67 |
| | S_3 | 0.10 | 0.07 | 0.74 | 1.03 | 0.52 | 0.63 | 21.73 | 4.38 | 0.38 |
| | S_4 | 0.68 | 0.32 | 0.55 | 21.14 | 2.23 | 0.17 | 954.22 | 10.92 | 0.02 |
| 50 | S_1 | 0.07 | 0.05 | 0.79 | 0.19 | 0.15 | 0.79 | 0.74 | 0.56 | 0.76 |
| | S_2 | 0.18 | 0.14 | 0.77 | 0.77 | 0.55 | 0.73 | 5.82 | 3.21 | 0.59 |
| | S_3 | 0.55 | 0.40 | 0.74 | 5.18 | 2.39 | 0.59 | 113.22 | 17.46 | 0.34 |
| | S_4 | 5.62 | 2.10 | 0.41 | 238.98 | 18.80 | 0.15 | 17598.39 | 128.65 | 0.02 |
| 100 | S_1 | 0.26 | 0.21 | 0.80 | 0.73 | 0.58 | 0.80 | 3.18 | 2.27 | 0.71 |
| | S_2 | 0.78 | 0.60 | 0.77 | 2.92 | 2.05 | 0.71 | 21.86 | 10.90 | 0.53 |
| | S_3 | 2.25 | 1.63 | 0.74 | 19.96 | 9.04 | 0.56 | 429.61 | 55.63 | 0.27 |
| | S_4 | 21.14 | 7.81 | 0.43 | 897.18 | 59.59 | 0.10 | 59093.25 | 516.44 | 0.02 |

Table 3: The average of counting time (T_1 for benchmark algorithm and T_2 for Algorithm 2) and ratios (T_2/T_1) for UCCGs with p vertices and pr edges in different subsets.

chordal graph is introduced and the size formula derivation based on the core graph is proposed. We discuss both recursive and explicit forms of the size formulas and give algorithm to derive these formulas. Comparing to the benchmark counting algorithm, the proposed algorithm can generate more size formulas efficiently, and by these formulas, size counting is accelerated dramatically when the essential graph contains non-sparse undirected subgraphs.

Acknowledgments

This work was supported partially by NSFC (11671020, 11101008, 71271211).

Appendix A. Algorithm ChainCom(\mathcal{U}, v)

For the completion of the paper, we give the algorithm ChainCom(\mathcal{U}, v) in Algorithm 3, which is introduced in He et al. (2015), to construct the rooted essential graph $\mathcal{U}^{(v)}$ and all of its chain components.

Appendix B. Proofs of Results

In this section, we provide the proofs of the main results of our paper.

Proof of Proposition 3

Let $v_{i_1} - v_{j_1}$ and $v_{i_2} - v_{j_2}$ be two edges in \mathcal{K}^c . If neither they share a common vertex, nor they are connected by an edge, we have that $v_{i_1}, v_{j_1}, v_{i_2}, v_{j_2}$ are four distinct vertices and there is no edge between v_{i_1}, v_{j_1} and v_{i_2}, v_{j_2} . Since that $\bar{\mathcal{K}}$ is the complement of \mathcal{K} ,

Algorithm 3: *ChainCom*(\mathcal{U}, v)**Input:** \mathcal{U} , a UCCG; v , a vertex of \mathcal{U} .**Output:** v -rooted essential graph of \mathcal{U} and all of its chain components.

```

1 Set  $A = \{v\}$ ,  $B = \tau \setminus v$ ,  $\mathcal{G} = \mathcal{U}$  and  $\mathcal{O} = \emptyset$ 
2 while  $B$  is not empty do
3   Set  $T = \{w : w \text{ in } B \text{ and adjacent to } A\}$ ;
4   Orient all edges between  $A$  and  $T$  as  $c \rightarrow t$  in  $\mathcal{G}$ , where  $c \in A, t \in T$ ;
5   repeat
6     for each edge  $y - z$  in the vertex-induced subgraph  $\mathcal{G}_T$  do
7       if  $x \rightarrow y - z$  in  $\mathcal{G}$  and  $x$  and  $z$  are not adjacent in  $\mathcal{G}$  then
8         Orient  $y - z$  to  $y \rightarrow z$  in  $\mathcal{G}$ 
9   until no more undirected edges in  $\mathcal{G}_T$  can be oriented;
10  Set  $A = T$  and  $B = B \setminus T$ ;
11  Append all isolated undirected graphs in  $\mathcal{G}_T$  to  $\mathcal{O}$ ;
12 return  $\mathcal{G}$  and  $\mathcal{O}$ 

```

we have that the four edges, $v_{i_1} - v_{i_2}$, $v_{i_2} - v_{j_1}$, $v_{j_1} - v_{j_2}$, and $v_{j_2} - v_{i_1}$ appear in \mathcal{K} , and meanwhile, the two edges $v_{i_1} - v_{j_1}$ and $v_{i_2} - v_{j_2}$ do not occur in \mathcal{K} . This implies that no chord exists in the cycle $v_{i_1} - v_{i_2} - v_{j_1} - v_{j_2} - v_{i_1}$ in \mathcal{K} . It is a contradiction because \mathcal{K} is a chordal graph.

Since no dominating vertices appear in \mathcal{K} , for any vertex v in \mathcal{K} , there exists another vertex in \mathcal{K} such that it is not adjacent to v . Consequently, there is no isolated vertex in \mathcal{K}^c . Following the proof in the last paragraph, there are no two edges that occur separately in two isolated subgraphs of \mathcal{K} . As a result, \mathcal{K}^c is a connected graph. ■

Before proving Theorem 5, we give the following lemma.

Lemma 11 *Let \mathcal{U} be an undirected chordal graph over V and $\mathcal{U}^{(v)}$ be the v -rooted graph of \mathcal{U} . We have that the subgraph of $\mathcal{U}^{(v)}$ on the neighbors of v , denoted by $\mathcal{U}_{N_v}^{(v)}$, is undirected.*

Proof We can get $\mathcal{U}^{(v)}$ using Algorithm 3. Consider any edge, denoted by $v_i - v_j$, in \mathcal{U}_{N_v} , v, v_i and v_j form a triangle. According to Algorithm 3, $v_i - v_j$ can not be oriented to a directed edge since $v \rightarrow v_i - v_j$ is not a induced subgraph of \mathcal{U} . Therefore, we have that $\mathcal{U}_{N_v}^{(v)}$ is undirected. ■

Proof of Theorem 5

Denote the vertices of \mathcal{K} as $V = \{v_1, \dots, v_p\}$, and the m extended vertices in \mathcal{K}^{m+} as $V' = \{v_{p+1}, \dots, v_{p+m}\}$. From Lemma 1, we have

$$f(\mathcal{K}, m) = \sum_{v \in V} \text{Size} \left((\mathcal{K}^{m+})^{(v)} \right) + \sum_{v \in V'} \text{Size} \left((\mathcal{K}^{m+})^{(v)} \right). \quad (10)$$

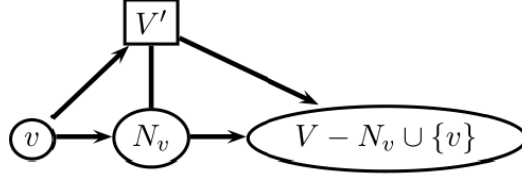


Figure 3: The directions of edges among v , N_v and V' and the other vertices in $(\mathcal{K}^{m+})^{(v)}$, where $v \rightarrow N_v$ represents that each edge between v and N_v is directed from v to the vertex in N_v , and $V' - N_v$ represents that all edges between V' and N_v are undirected.

For any $v \in V'$, since v is adjacent to all other vertices in \mathcal{K}^{m+} , from Lemma 11, we have $\text{Size}((\mathcal{K}^{m+})^{(v)}) = f(\mathcal{K}, m - 1)$ and

$$\sum_{v \in V'} \text{Size}((\mathcal{K}^{m+})^{(v)}) = m \cdot f(\mathcal{K}, m - 1). \quad (11)$$

For any $v \in V$, the neighbor set of v in $\mathcal{K}^{(m+)}$ is $N_v \cup V'$, from Lemma 11, $(\mathcal{K}_{N_v})^{m+}$ is a chain component of $(\mathcal{K}^{m+})^{(v)}$ when $m > 0$. According to Algorithm 3 and Lemma 11, the directions of edges among v , N_v and V' and the other vertices in $(\mathcal{K}^{m+})^{(v)}$ are displayed in Figure 3. All edges are directed from $N_v \cup V'$ to $V - N_v \cup \{v\}$ in $(\mathcal{K}^{m+})^{(v)}$. We have

$$\text{Size}((\mathcal{K}^{m+})^{(v)}) = \text{Size}\left(\left((\mathcal{K}^{m+})^{(v)}\right)_{N_v \cup V'}\right) \text{Size}\left(\left((\mathcal{K}^{m+})^{(v)}\right)_{V - N_v \cup \{v\}}\right)$$

First, according to Lemma 11, we can get that $((\mathcal{K}^{m+})^{(v)})_{N_v \cup V'}$ is the same as $(\mathcal{K}_{N_v})^{m+}$, thus, $\text{Size}\left(\left((\mathcal{K}^{m+})^{(v)}\right)_{N_v \cup V'}\right) = f(\mathcal{K}_{N_v}, m)$ holds. Then, consider the undirected edges in $((\mathcal{K}^{m+})^{(v)})_{V - N_v \cup \{v\}}$, according to Algorithm 3, because all vertices in V' are parents of vertices in $V - N_v \cup \{v\}$, we have that $((\mathcal{K}^{m+})^{(v)})_{V - N_v \cup \{v\}}$ has the same chain components as $(\mathcal{K}^{(v)})_{V - N_v \cup \{v\}}$. As a result, $\text{Size}\left(\left((\mathcal{K}^{m+})^{(v)}\right)_{V - N_v \cup \{v\}}\right) = \text{Size}\left(\left(\mathcal{K}^{(v)}\right)_{V - N_v \cup \{v\}}\right)$. Moreover, according to Equation (1), we have $\text{Size}\left(\left(\mathcal{K}^{(v)}\right)_{V - N_v \cup \{v\}}\right) = \frac{\text{Size}(\mathcal{K}^{(v)})}{\text{Size}(\mathcal{K}_{N_v})}$. Consequently, we have

$$f((\mathcal{K}^{m+})^{(v)}) = f(\mathcal{K}_{N_v}, m) \frac{\text{Size}(\mathcal{K}^{(v)})}{\text{Size}(\mathcal{K}_{N_v})}. \quad (12)$$

Theorem 5 holds directly from Equation (10), Equation (11) and Equation (12). ■

Proof of Corollary 6

For any undirected chordal graph \mathcal{K} , from Theorem 5, we have

$$f(\mathcal{K}, m) = m \cdot f(\mathcal{K}, m - 1) + \sum_{v \in V} f(\mathcal{K}_{N_v}, m) h(\mathcal{K}, v) \quad (13)$$

where V is the set of vertices in \mathcal{K} , $h(\mathcal{K}, v)$ is an integer function of \mathcal{K} and v . Consider $f(\cdot, \cdot)$ terms in the right side of Equation (13), we can calculate them by using Equation (13) again as follows.

$$f(\mathcal{K}, m-1) = (m-1) \cdot f(\mathcal{K}, m-2) + \sum_{v \in V} f(\mathcal{K}_{N_v}, m-1) h(\mathcal{K}, v) \quad (14)$$

and

$$f(\mathcal{K}_{N_v}, m) = m \cdot f(\mathcal{K}_{N_v}, m-1) + \sum_{v' \in N_v} f(\mathcal{K}_{N'_{v'}}, m) h(\mathcal{K}_{N_v}, v), \quad (15)$$

where $N'_{v'}$ is the neighbor set of v' in \mathcal{K}_{N_v} . Replacing $f(\mathcal{K}, m-1)$ and $f(\mathcal{K}_{N_v}, m)$ in Equation (13) by the corresponding terms in Equation (14) and Equation (15), we can find that $f(\mathcal{K}, m)$ is the sum of the following three types of terms,

1. $m(m-1)f(\mathcal{K}, m-2)$,
2. $mf(\mathcal{K}_{N_v}, m-1)h(\mathcal{K}, v)$, for any $v \in V(\mathcal{K})$, and
3. $f(\mathcal{K}_{N'_{v'}}, m)h(\mathcal{K}_{N_v}, v)$, where $N'_{v'}$ is the neighbor set of v' in \mathcal{K}_{N_v} , for any v', v such that $v' \in N_v$.

Notice that for any v, v' , we have $N'_{v'} \subset N_v \subset V$, so the graphs in above three types of terms are smaller than that in Equation (13). By this way, using Equation (13) repeatedly, we can calculate $f(\mathcal{K}, m)$ by smaller graphs. Finally, $f(\mathcal{K}, m)$ can be calculated only by $f(\mathcal{K}, 0)$ and $f((\mathcal{K}_\emptyset), k)$ for $k \leq m$. As a result, $f(\mathcal{K}, m)$ is the sum of some polynomials of m and each term of the polynomials contains either $m!f(\mathcal{K}, 0)$ or $\frac{m!}{k!}f((\mathcal{K}_\emptyset), k)$ for $k \leq m$. Because \mathcal{K}_\emptyset is null graph, $f((\mathcal{K}_\emptyset), k) = k!$, we have that $f(\mathcal{K}, m)$ is a polynomial divisible by $m!$. \blacksquare

Proof of Theorem 7

We just need to show that Formula (7) is the solution of Equation (4). First, when $m = 0$, we have $f(\mathcal{K}, m) = \beta_0 = \text{Size}(\mathcal{K})$. Theorem 7 holds if the following equation holds,

$$\left(\beta_0 + \sum_{i=1}^{d+1} \beta_i m^i \right) m! = m \left(\beta_0 + \sum_{i=1}^{d+1} \beta_i (m-1)^i \right) (m-1)! + \sum_{i=1}^{d+1} \gamma_i m^{i-1} m!.$$

Equivalently,

$$\sum_{i=1}^{d+1} \beta_i m^i - \sum_{i=1}^{d+1} \beta_i (m-1)^i = \sum_{i=1}^{d+1} \gamma_i m^{i-1} \quad (16)$$

Consider the left side of Equation (16),

$$\begin{aligned} \sum_{i=1}^{d+1} \beta_i [m^i - (m-1)^i] &= \sum_{i=1}^{d+1} \beta_i \left[\sum_{j=0}^{i-1} (-1)^{i-(j+1)} \binom{i}{j} m^j \right] \\ &= \sum_{j=0}^d \sum_{i=j+1}^{d+1} \left[(-1)^{i-(j+1)} \binom{i}{j} \beta_i m^j \right] \\ &= \sum_{k=1}^{d+1} \left[\sum_{i=k}^{d+1} (-1)^{i-k} \binom{i}{k-1} \beta_i \right] m^{k-1} \end{aligned}$$

If Equation (16) holds for any $m > 0$, we have that $\sum_{i=k}^{d+1} (-1)^{i-k} \binom{i}{k-1} \beta_i = \gamma_k$ holds for any $k = 1, \dots, d+1$. Let

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1,d+1} \\ 0 & a_{22} & \cdots & a_{2,d+1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{d+1,d+1} \end{pmatrix} = \begin{pmatrix} \binom{1}{0} & -\binom{2}{0} & \cdots & (-1)^d \binom{d+1}{0} \\ 0 & \binom{2}{1} & \cdots & (-1)^{d-1} \binom{d+1}{1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \binom{d+1}{d} \end{pmatrix}$$

and $\beta = (\beta_1, \dots, \beta_{d+1})^T$, and $\gamma = (\gamma_1, \dots, \gamma_{d+1})^T$. We have

$$A\beta = \gamma.$$

It is easy to verify that β in Theorem 7 is the solution of $A\beta = \gamma$. ■

Proof of Corollary 8

Let v_1, \dots, v_j be the j isolated vertices, v'_1, \dots, v'_m be the m extended vertices. $V(\mathcal{K}_1)$ be the vertices in \mathcal{K}_1 , $V(\mathcal{K})$ be the vertices in \mathcal{K} . Clearly, we have $V(\mathcal{K}) = V(\mathcal{K}_1) \cup \{v_1, \dots, v_j\}$

Because \mathcal{K} is composed of \mathcal{K}_1 and j isolated vertices, Equation (8) holds when $m = 0$ since $\text{Size}(\mathcal{K}) = \text{Size}(\mathcal{K}_1)$. Consider the case $m = 1$. From Theorem 5, we have

$$f(\mathcal{K}, m) = m \cdot f(\mathcal{K}, m-1) + \sum_{v \in V(\mathcal{K})} f(\mathcal{K}_{N_v}, m) \frac{\text{Size}(\mathcal{K}^{(v)})}{\text{Size}(\mathcal{K}_{N_v})}. \quad (17)$$

Since $m-1 = 0$, we have $f(\mathcal{K}, m-1) = \text{Size}(\mathcal{K}) = \text{Size}(\mathcal{K}_1)$, and $m \cdot f(\mathcal{K}, m-1) = m \cdot f(\mathcal{K}_1)$. Moreover, for any $v \in V(\mathcal{K}_1)$, $\text{Size}(\mathcal{K}^{(v)}) = \text{Size}((\mathcal{K}_1)^{(v)})$ and $\mathcal{K}_{N_v} = (\mathcal{K}_1)_{N_v}$ hold. For any $v \in \{v_1, \dots, v_j\}$, $\text{Size}(\mathcal{K}^{(v)}) = \text{Size}(\mathcal{K}_1)$ and \mathcal{K}_{N_v} is a null graph; it follows $f(\mathcal{K}_{N_v}, m) = m!$ and $\text{Size}(\mathcal{K}_{N_v}) = 1$. From Equation (17), we have that

$$\begin{aligned} f(\mathcal{K}, m) &= m \cdot \text{Size}(\mathcal{K}_1) + \sum_{v \in V(\mathcal{K}_1)} f((\mathcal{K}_1)_{N_v}, m) \frac{\text{Size}(\mathcal{K}_1^{(v)})}{\text{Size}((\mathcal{K}_1)_{N_v})} + \text{Size}(\mathcal{K}_1)jm! \\ &= f(\mathcal{K}_1, m) + \text{Size}(\mathcal{K}_1)jm! = f(\mathcal{K}_1, 1) + \text{Size}(\mathcal{K}_1)j \end{aligned}$$

We have Equation (8) holds for $m = 1$. Suppose that Equation (8) holds for $m = k-1$, consider $m = k$, from Equation (17), we have

$$f(\mathcal{K}, k) = k \cdot f(\mathcal{K}, k-1) + \sum_{v \in V(\mathcal{K}_1)} f((\mathcal{K}_1)_{N_v}, k) \frac{\text{Size}(\mathcal{K}_1^{(v)})}{\text{Size}((\mathcal{K}_1)_{N_v})} + \text{Size}(\mathcal{K}_1)jk! \quad (18)$$

Since Equation (8) holds for $m = k-1$, we have

$$f(\mathcal{K}, k-1) = f(\mathcal{K}_1, k-1) + j(k-1)\text{Size}(\mathcal{K}_1)(k-1)!. \quad (19)$$

From Theorem 5, we can get

$$f(\mathcal{K}_1, k) = kf(\mathcal{K}_1, k-1) + \sum_{v \in V(\mathcal{K}_1)} f((\mathcal{K}_1)_{N_v}, k) \frac{\text{Size}(\mathcal{K}_1^{(v)})}{\text{Size}((\mathcal{K}_1)_{N_v})}. \quad (20)$$

From Equation (18), (19), and (20), we have

$$\begin{aligned} f(\mathcal{K}, k) &= f(\mathcal{K}_1, k) + j(k-1)\text{Size}(\mathcal{K}_1)k! + \text{Size}(\mathcal{K}_1)jk! \\ &= f(\mathcal{K}_1, k) + \text{Size}(\mathcal{K}_1)jkk!. \end{aligned}$$

As a result, Equation (8) holds for any integer $m \geq 0$. ■

Proof of Corollary 9

The proof of (1)

When \mathcal{K} is null graph, \mathcal{K}^{m+} is a completed graph with m vertices, the result (1) holds obviously.

The proof of (2)

Let $d_1, d_2, d_3, \dots, d_p$ be degrees of vertices v_1, \dots, v_p in \mathcal{K} , we have $\sum_{i=1}^p d_i = 2(p-1)$. Consider $g(\mathcal{K}, m)$ defined in Equation (5),

$$g(\mathcal{K}, m) = \sum_{i=1}^p \frac{f(\mathcal{K}_{N_{v_i}}, m)}{m!} \frac{\text{Size}(\mathcal{K}^{(v_i)})}{\text{Size}(\mathcal{K}_{N_{v_i}})}.$$

Since \mathcal{K} is a tree, we have that $\mathcal{K}_{N_{v_i}}$ is composed of d_i isolated vertices, so $\frac{f(\mathcal{K}_{N_{v_i}}, m)}{m!} = 1 + d_i m$. We also have $f(\mathcal{K}^{(v_i)}) = 1$ and $f(\mathcal{K}_{N_{v_i}}) = 1$ if \mathcal{K} is a tree. Consequently,

$$g(\mathcal{K}, m) = \sum_{i=1}^p (1 + d_i m) = p + 2(p-1)m$$

The result (2) holds according to Theorem 7.

The proof of (3)

Consider a tree plus graph, if it is chordal, the added edge must be in a triangle, otherwise, the tree plus graph is not chordal. Let $d_1, d_2, d_3, \dots, d_p$ be degrees of vertices v_1, \dots, v_p in \mathcal{K} and d_1, d_2, d_3 are the degrees of the three vertices in the triangle, we have $\sum_{i=1}^p d_i = 2p$. Moreover, considering the induced subgraph of \mathcal{K} over N_{v_i} , we have that $\mathcal{K}_{N_{v_i}}$ is composed of an edge and $d_i - 2$ isolated vertices for $i = 1, 2, 3$, and $\mathcal{K}_{N_{v_i}}$ just contains d_i isolated vertices for $i = 4, 5, \dots, p$. Following Corollary 8, we can calculate $g(\mathcal{K}, m)$ as following

$$\begin{aligned} g(\mathcal{K}, m) &= \frac{1}{m!} \left[2(d_1 + d_2 + d_3 - 6)mm! + 3(m+2)! + 2 \sum_{i \neq 1, 2, 3} (d_i mm! + m!) \right] \\ &= 3m^2 + 4pm - 3m + 2p \end{aligned}$$

The result (3) holds according to Theorem 7.

The proof of (4)

Consider a vertex v in \mathcal{K} , we have that $(\mathcal{K}^{m+})^{(v)}$ contains $p/2$ chain components, in which one is a completed graph with $m+1$ vertices, and the others are one-edge graphs. We can calculate $g(\mathcal{K}, m)$ defined in Equation (6) as following

$$g(\mathcal{K}, m) = 2^{p/2}pm/2 + 2^{p/2}p/2.$$

As a result, the result (4) holds according to Theorem 7. ■

Proof of Corollary 10

According to the definition of $g(\mathcal{K}, m)$ in Equation (5)

$$g(\mathcal{K}, m) = \sum_{j=1}^J \sum_{v \in V(\mathcal{K}_j)} \frac{f(\mathcal{K}_{N_v}, m)}{m!} \frac{\text{Size}(\mathcal{K}^{(v)})}{\text{Size}(\mathcal{K}_{N_v})}.$$

Because \mathcal{K} is composed of $\mathcal{K}_1, \dots, \mathcal{K}_J$ that are J isolated connected graphs, we have that $\mathcal{K}_{N_v} = \mathcal{K}_{j, N_v}$, and $\text{Size}(\mathcal{K}^{(v)}) = \text{Size}(\mathcal{K}_j^{(v)}) \prod_{l \neq j} \text{Size}(\mathcal{K}_l) = \text{Size}(\mathcal{K}_j^{(v)}) \frac{\text{Size}(\mathcal{K})}{\text{Size}(\mathcal{K}_j)}$. Consequently, Corollary 10 holds. ■

References

- S. A. Andersson, D. Madigan, and M. D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *The Annals of Statistics*, 25(2):505–541, 1997.
- D. M. Chickering. Learning equivalence classes of Bayesian-network structures. *The Journal of Machine Learning Research*, 2:445–498, 2002.
- S. B. Gillispie. Formulas for counting acyclic digraph Markov equivalence classes. *Journal of Statistical Planning and Inference*, 136(4):1410–1432, 2006.
- S.B. Gillispie and M.D. Perlman. The size distribution for Markov equivalence classes of acyclic digraph models. *Artificial Intelligence*, 141(1-2):137–155, 2002.
- Yangbo He and Zhi Geng. Active learning of causal networks with intervention experiments and optimal designs. *Journal of Machine Learning Research*, 9:2523–2547, 2008.
- Yangbo He, Jinzhu Jia, and Bin Yu. Counting and exploring sizes of markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 16:2589–2609, 2015.
- M. H. Maathuis, M. Kalisch, and P. Bühlmann. Estimating high-dimensional intervention effects from observational data. *The Annals of Statistics*, 37(6A):3133–3164, 2009. ISSN 0090-5364.
- J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge Univ Pr, 2000.
- R. Robinson. Counting labeled acyclic digraphs. *New Directions in the Theory of Graphs*, pages 239–273, 1973.
- R. Robinson. Counting unlabeled acyclic digraphs. In *Combinatorial mathematics V*, pages 28–43. Springer, 1977.
- P. Spirtes, C.N. Glymour, and R. Scheines. *Causation, Prediction, and Search*. The MIT Press, 2001.
- B. Steinsky. Enumeration of labelled chain graphs and labelled essential directed acyclic graphs. *Discrete mathematics*, 270(1-3):266–277, 2003.

- T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, page 270. Elsevier Science Inc., 1990.